Summary

I. Objective

To summarize the numerical techniques presented in this course and to discuss the history behind the invention of the digital computer.

II. Final Comments

Next week, on Friday April 26, we will give you a list of final projects to choose from. You will pick one problem and write a FORTRAN program to solve it. Your final project will be due on Friday May 3 by 5 pm. No work will be graded after this date. The recursion relations for all the numerical techniques presented in this course are summarized in the Appendix. You may find this summary useful when you begin writing your programs next week.

In this course we have seen many physics applications of the computer, from data taking and analysis to the solution of complex mathematical equations. It is interesting to note that the world's first digital computer was built in a workshop just a few feet from the room where this class meets today.

The first computer to perform arithmetic electronically was invented in the basement of ISU's physics building[1] by Dr. J. V. Atanasoff (pronounced "at-an-**A**-soff")and Clifford Berry. Completed in 1942, the ABC (short for Atanasoff-Berry Computer) contained 300 vacuum tubes and could solve a system of simultaneous equations as large as 29x29. Several ideas developed by Atanasoff are still in use today, such as performing calculations in base 2. Dr. Lester Earls, an ISU physics department faculty member who met Atanasoff during the 1930's, described chatting with Atanasoff in his office. The office has since been converted into a classroom (room 52 on the southeast end of the old physics building), and the basement workshop where the computer was built is now used as a storage area. The article at the end of this handout describes the development of the ABC. Read it and gain an appreciation for the contributions of Atanasoff and Berry, as well as for how far computers have come in a mere sixty years.

---

[1] R. Slater, *Computers in Physics,* 44 (Nov/Dec 1987). See the article *Who Invented the Computer?* at the back of this handout.

III. Appendix: Formulas for Numerical Algorithms[2]

A. Notation

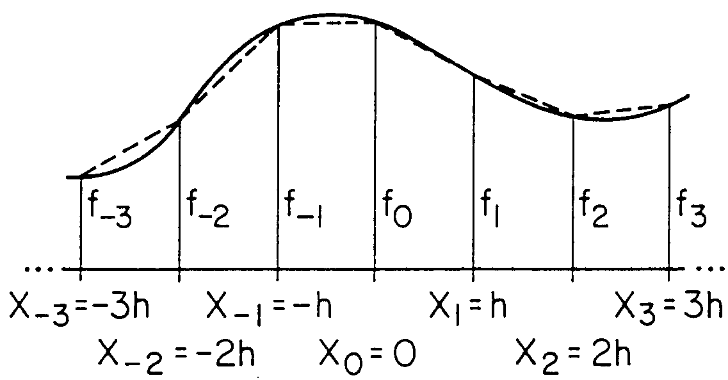
The following formulas are all based on the assumption that the function f(x) is defined on a number line of equally spaced coordinates. We define the following notation:

$$f_n \equiv f(x_n) \text{ where } x_n \equiv nh \; (n = 0, \pm 1, \pm 2, etc.)$$

The table below tells you how to think about this.

Think of ...	...as value of the function evaluated...
f_0	at the point of interest
f_1, f_{-1}	one lattice space to the right / left of f_0
f_2, f_{-2}	two lattice spaces to the right / left of f_0

The figure below illustrates this.



B. Differentiation

First Derivative Formulas			
Method Name	**Formula**	**Error**	**Reference**
3 point formula	$f' = \frac{1}{2h}[f_1 - f_{-1}]$	$O(h^2)$	Koonin p.5
5 point formula	$f' = \frac{1}{12h}[f_{-2} - 8f_{-1} + 8f_1 - f_2]$	$O(h^4)$	Koonin p.5
Second Derivative Formulas			
Method Name	**Formula**	**Error**	**Reference**
3 point formula	$f'' = \frac{1}{h^2}[f_1 - 2f_0 + f_{-1}]$	$O(h^2)$	Koonin p.5
5 point formula	$f'' = \frac{1}{12h^2}[-f_{-2} + 16f_{-1} - 30f_0 + 16f_1 - f_2]$	$O(h^4)$	Koonin p.6

[2] All of the differentiation and integration formulas are derived in the book *Computational Physics* by Steven Koonin which is on reserve in Parks Library. The formulas are obtained by writing the Taylor expansion of the function

$$f(x) = f_0 + xf' + \frac{x^2}{2!}f'' + \frac{x^3}{3!}f''' + \ldots$$

where all the derivatives are evaluated x = 0.

C. Integration

We will break up integrals over the range [a,b] into the smaller integrals (for Trapezoidal and Simpson's Rule)

$$\int_a^b f(x)dx = \int_a^{a+2h} f(x)dx + \int_{a+2h}^{a+4h} f(x)dx + ... + \int_{b-2h}^{b} f(x)dx$$

or (for Bode's Rule)

$$\int_a^b f(x)dx = \int_a^{a+4h} f(x)dx + \int_{a+4h}^{a+8h} f(x)dx + ... + \int_{b-4h}^{b} f(x)dx$$

and apply the formulas for the smaller integrals given below. Note that the interval length h is $h = \frac{b-a}{N}$ where N is the number of intervals. Note from the table that N must be chosen to be a multiple of 2 or 4 for the methods below.

Integration Formulas				
Method Name	**Formula**	**Error**	**N**	**Reference**
Trapezoidal Rule	$\int_{-h}^{h} f(x)dx = \frac{h}{2}(f_{-1} + 2f_0 + f_1)$	$O(h^3)$	must be a multiple of 2	Koonin p.6
Simpson's Rule	$\int_{-h}^{h} f(x)dx = \frac{h}{3}[f_1 + 4f_0 + f_{-1}]$	$O(h^5)$	must be a multiple of 2	Koonin p.6
Bode's Rule	$\int_0^{4h} f(x)dx = \frac{2h}{45}[7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4]$	$O(h^7)$	must be a multiple of 4	Koonin p.8

D. Root Solving

Method Name	Recursion Formula	Reference
Secant Method	$x_{i+1} = x_i - f(x_i)\left(\frac{x_i - x_{i-1}}{f(x_i) - f(x_{i-1})}\right)$	Koonin p.12

E. First Order Differential Equations

1. Uncoupled Equations

The table below gives recursive formulas for solving differential equations of the form

$$\frac{dy}{dx} = f(x,y)$$

given the value of *y* at *x* =0. The symbol *f(x,y)* denotes a function of *x* (the independent variable) and *y* (the dependent variable). We define the step size $h \equiv x_{n+1} - x_n$ and the symbol $y_n \equiv y(x_n)$. The term **global error** in the table refers to the error in y_n at the final x value. The notation O(h) means that the error is on the order of the value h.

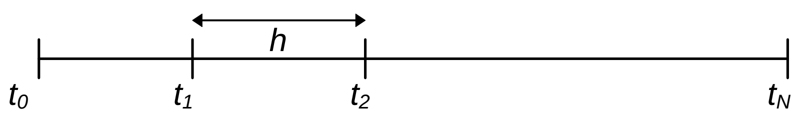
Method Name	Formula	Global Error	Reference
Euler's Method	$y_{n+1} = y_n + hf(x_n, y_n)$	O(h)	Koonin p.6
Runge-Kutta Method[3]	$y_{n+1} = y_n + hf(x_n + \frac{h}{2}, y_n + \frac{k}{2})$ $k = hf(x_n, y_n)$	O(h^2)	Koonin p.6

2. Coupled Equations

Coupled equations of the form

$$\frac{dx}{dt} = X(x,y,z)$$
$$\frac{dy}{dt} = Y(x,y,z)$$
$$\frac{dz}{dt} = Z(x,y,z)$$

may be numerically solved for $x(t)$, $y(t)$, and $z(t)$ using the **Euler Method**[4] if the initial conditions $x(0)$, $y(0)$, and $z(0)$) are known. The t axis is divided into N intervals of equal length $h = \frac{t_N - t_0}{N}$ as shown below. This method may be summarized by the recursion



relations in the table below. We use the notation $x_i \equiv x(t_i)$, and similar notation for y and z.

Euler Method Recursion Relations for *x, y,* and *z*
$x_0 \equiv x(0)$ $x_{i+1} = x_i + hX(x_i, y_i, z_i)$
$y_0 \equiv y(0)$ $y_{i+1} = y_i + hY(x_i, y_i, z_i)$
$z_0 \equiv z(0)$ $z_{i+1} = z_i + hZ(x_i, y_i, z_i)$

[3] There are many different Runge-Kutta algorithms. See Koonin for methods of order 3 and higher.

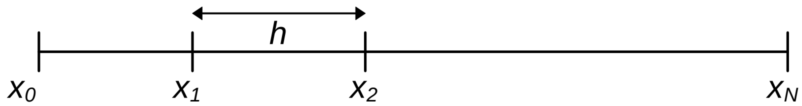
[4] See S. E. Koonin, *Computational Physics,* Chapter 2, for a complete description.

F. Second Order Differential Equations

Equations of the form

$$\frac{d^2y}{dx^2}+k^2(x)y=S(x)$$

may be solved for $y(x)$ for points on the axis using the **Numerov Algorithm**[5] if the boundary conditions $y(x_0)$, $y'(x_0)$ are known at some point x_0.. The axis is divided into N intervals of equal length $h=\frac{x_N-x_0}{N}$ as shown below. This method may be summarized by the recursion



relations in the table below. We use the notation $y_i \equiv y(x_i)$, $y_i' = y'(x_i)$ and similar notation for k_i and S_i.

Numerov Algorithm Recursion Relations for *y_0, y_1,* and *y_i*
$y_0 = y(x_0)$
$y_1 = y_0 + y_0' h + \frac{(-k_0^2 y_0 + S_0)}{2} h^2$
$y_i = \frac{1}{\left(1+\frac{h^2k_i^2}{12}\right)}\left\{\frac{h^2}{12}\left(S_i+10S_{i-1}+S_{i-2}\right)+2\left(1-\frac{5h^2k_{i-1}^2}{12}\right)y_{i-1}-\left(1+\frac{h^2k_{i-2}^2}{12}\right)y_{i-2}\right\}$

[5] See S. E. Koonin, *Computational Physics,* Chapter 3, for a complete description.